AMENDMENTS TO THE SPECIFICATION

Please replace the paragraphs beginning on page 1, line 10 and ending on page 2, line 10 with the following paragraphs:

CROSS-REFERENCE TO RELATED APPLICATIONS

A1 This application claims priority of the following United States provisional applications, all of which are incorporated herein by reference: Serial Number 60/255,044, filed December 11, 2000, titled Framework for Web-Based Integration of Management Applications; Serial Number 60/255,051, filed December 11, 2000, titled Schema Driven User Task Architecture; Serial Number 60/255,238, filed December 11, 2000, titled Schema Driven Property Pages; Serial Number 60/255,041, filed December 11, 2000, titled Schema Driven Search; Serial Number 60/255,153, filed December 11, 2000, titled Search Engine Extensibility; Serial Number 60/255,043, filed December 11, 2000, titled Management Portal Explorer; Serial Number 60/255,050, filed December 11, 2000, titled Object Property Representation; [[_____;]] Serial Number 60/255,042, filed December 11, 2000, titled Quick Search; [[_____;]] and Serial Number 60/255,052, filed December 11, 2000, titled Mmp Modules – Query Driven Model For Location Objects and Tasks[[_____]].

This application is related to the following commonly assigned United States patent applications, all of which were filed concurrently with this application, and all of which are incorporated herein by reference: Serial Number 10/014,120 [[_____,]] ~~(Attorney Docket Number 40062.153USU1/MS167417.2)~~, filed December 11, 2001, titled User Interface for Managing Multiple Network Resources; Serial Number 10/014,177 [[_____,]] ~~(Attorney Docket Number 40062.147USU1/MS167411.2)~~, filed December 11, 2001, titled Method and System for Management of Multiple Network Resources; Serial Number 10/014,114 [[_____,]] ~~(Attorney Docket Number 40062.150USU1/MS167414.2)~~, filed December 11, 2001, titled System and Method for Representing an Object Used in Management of Multiple Network Resources; Serial Number 10/014,293 [[_____,]] ~~(Attorney Docket Number 40062.148USU1/MS167412.2)~~, filed December 11, 2001, titled Method and System for Task-Based Management of Multiple Network Resources; and Serial Number

10/015,050 [[_____,]] ~~(Attorney Docket Number 40062.152USU1/MS167416.2)~~, filed December 11, 2001, titled Navigation Tool for Accessing Workspaces and Modules in a Graphical User Interface.

A1

Please replace the paragraph beginning on page 2, line 19 and ending on page 3 line 4, with the following rewritten paragraph:

A2

Network administrators, also known as information technology staff, perform many operations throughout each day in order to manage a number of different resources associated with their respective networks. These networks may comprise any number of hardware related computer resources, e.g., printers, computer stations, servers, etc., as well as a number of software related resources, such as databases, employee profiles, email servers, and applications, among others. Typically, each of these resources has a uniquely different front end or user interface that the administrator must use to modify, evaluate or otherwise configure that resource. Consequently, in order to perform their duties, administrators must be familiar with many different types of systems.

Please replace the paragraph beginning on page 6, line 11 and ending on page 7, line 3 with the following rewritten paragraph:

A3

In accordance with yet other aspects, the present invention relates to a system and method ~~involving~~ that associates a first search component with a first object type, wherein the first search component relates to object attributes managed by a first resource; and associates a second search component with the first object type, wherein the second search component relates to object attributes managed by a second resource. Consequently, in a network environment having multiple resources, search information may be shared between different resources. The system may include a management module in communication with the plurality of resources, the management module capable of receiving a request to access information related to one or more of the plurality of resources and to receive search information from the plurality of resources, the request relating to a query; and in response to the receipt of a query request, the management module performing search functions on more than one resource to display accessed information. The management module may utilize a search manager to receive and store search information, the search manager further communicates with the resources to perform the requested searches.

#3 Furthermore, each of the resources may include a dedicated search engine for performing searches.

Please replace the paragraph beginning on page 10, line 18 and ending on page 11, line 2 with the following rewritten paragraph:

#4 The resources 108 are operably connected to the server computer system 104 such that information can be sent to and received from the resources. These connections are common in distributed environments, and may be wireless or not. The protocols used to communicate between the resources and server computer system may be proprietary to the resource and/or the server 104. However, the protocol used should preferably allow for the ability to control various features of the resources, such as being able to modify the configuration of the resource, such as being able to update its database, change its security options, etc.

Please replace the paragraph beginning on page 11, line 3 and ending on page 11, line 9 with the following rewritten paragraph:

#5 The client computer system 102 may communicate with the server computer system 104 via many different protocols over various types of connections. As shown in Fig. 1, the systems 102 and 104 may communicate via the Internet 106 using Hypertext Transfer Protocol (HTTP), mark-up languages, or some other communication protocol suitable for use with, for example, the Internet. In a particular embodiment, client computer system 102 is a ~~Microsoft.NET~~ Microsoft®.NET client, but other, ~~non-Microsoft.NET~~ non-Microsoft®.NET clients may be used.

Please replace the paragraph beginning on page 13, line 4 and ending on page 13, line 20 with the following rewritten paragraph:

#6 In its most basic configuration, the computing system 200 is illustrated in Fig. 2 by the dashed line 206. Additionally, the system 200 may also include additional storage (removable and/or non-removable) including, but not limited to, magnetic or optical disks or tape. Such additional storage is illustrated in Fig. 2 by removable storage 208 and non-removable storage 210. Typically, the bulk of the database information is stored in such additional storage. Computer storage media includes volatile and nonvolatile, removable and non-removable media

implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules or other data. Memory 204, removable storage 208, and non-removable storage 210 are all examples of computer storage media. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can be accessed by the system 200. Any such computer storage media may be part of system 200. Depending on the configuration and type of computing device, the memory 204 may be volatile, non-volatile or some combination of the two.

Please replace the paragraph beginning on page 15, line 5 and ending on page 15, line 19 with the following rewritten paragraph:

Additionally, the client computer system software 308 also communicates with or provides functionality to the user through the user interface module 310. The user interface module 310 may employ a GUI, such as GUI 116 shown in Fig. 1, or may employ a command line interface. In one embodiment, discussed in detail below, the user interface includes a web browser application 311 to present the GUI 116. To facilitate the presentation of the GUI in the web browser 311, the interface module 310 may also include an applet 313. Details regarding both the web browser ~~313~~ 311 and the applet 313 are provided below. The user interface 310 may further include other elements such as input elements, e.g., keyboards, touch screens, touch pads, mice, among others. The user interface 310 may also include output elements such as the graphical user interface 116, as shown in Fig. 1, or other output elements such as printers, speakers, etc. An end user, such as a network administrator, communicates with the computer system software 308 through the user interface module 310. In turn, the computer system software 308 communicates with the management module 304 to effectively manage resources 306.

Please replace the paragraph beginning on page 17, line 7 and ending on page 17, line 12 with the following rewritten paragraph:

A8

Next, in this embodiment, a ~~Microsoft~~ Microsoft® ".NET Framework" assembly is created. The assembly for the plugin may implement the following interfaces: IPluginPropertySheet, IPluginScope, and IPluginSearch, which, in general, implement methods for managing property sheets, search functions and other elements of the plugin. In order to manage the plugin, these interfaces may be accessed by the system management environment 304.

Please replace the paragraph beginning on page 17, line 13 and ending on page 18, line 3 with the following rewritten paragraph:

A9

The management module 304, which may be present on a server computer system such as 104 shown in Fig. 1, provides a "middle tier" of the management system 100 shown in Fig. 1. In a particular embodiment, the environment 300 is based on the ~~Microsoft~~ Microsoft® ".NET" framework. The server-side functionality may potentially be written in C# and communicate via IIS and ASP+ and be hosted on a single server or on a web farm. The management module 304 may be located in a separate domain to the client and also to the managed resources and communication can take place between firewalls. In this embodiment, the managed resources can be written in any .NET compliant language, for example, C# or VB.NET. The managed resources preferably also expose certain web services to facilitate communication with the management module 304. Additionally, the management module 304 may be a web service associated with the .NET framework. Web Services are described in more detail in the article titled "The Programmable Web: Web Services Provides Building Blocks for ~~Microsoft~~ Microsoft® .NET Framework" from the magazine ~~"MSDN Magazine" September 2000 issue~~.

Please replace the paragraph beginning on page 18, line 4 and ending on page 18, line 15 with the following ~~rewritten~~ paragraph:

A10

The management module 304 uses many elements to facilitate management of the resources 306. In an embodiment of the invention, the management module 304 has a plurality of managers that operate relatively independently to perform various functions in managing the resources 306. Each manager may further have a store of information that resides on the computer system housing the manager itself, as discussed in more detail below. In an embodiment of the invention, the management module 304 has a search manager 324, a user

interface manager 326, a task manager 328, a configuration manager 330, a property sheet manager 332, and a persistence manager 334. In alternative embodiments the management module 304 may include other managers. Moreover, in yet other embodiments, the management module 304 may incorporate fewer managers than shown in Fig. 304. The various managers communicate with each other as indicated by connection 335 shown in Fig. 3.

A10

Please replace the paragraph beginning on page 18, line 23 and ending on page 19, line 9 with the following rewritten paragraph:

A11

With respect to the user interface manager 326, the manager 326 provides the interface functionality between the management module 304 and the client computer system software 308. In an embodiment, the user interface module 326 provides many advanced features, e.g., the user interface manager 326 may customize output information for a particular client computer system 302. Consequently, the user interface manager 326 allows for the use of many different types of client computer systems 302, e.g., laptops, desktops, PDAs, cell phones, etc. The user interface manager 326 communicates with the client computer system 302 to provide the proper format and amount of output information, as well as input information. Furthermore, the user interface manager 326 may adapt to many different network protocols which may be used across the distributed network.

Please replace the paragraph beginning on page 19, line 13 and ending on page 19, line 16 with the following rewritten paragraph:

A12

In general, the persistence manager 334 functions to store various predefined and authored layouts of a management console graphical user interface (console layouts) that is described in detail below. The persistence manager 334 preferably stores these console layouts as XML type files in a memory, such as persistence store 346, shown in Fig. 3.

Please replace the paragraph beginning on page 19, line 17 and ending on page 19, line 22 with the following rewritten paragraph:

A13

With respect to the search manager 324, it generally parses and performs search operations on portions of data stored and used on the system 300. Typically, the search manager 324 receives a query from the client computer system software 308, and performs the query

A13 using information stored in search store 336 as well as through functional interaction with the back-end resources 306 to thereby supply search results back to the client 308.

Please replace the paragraph beginning on page 19, line 23 and ending on page 20 line 14 with the following rewritten paragraph:

A14 Similarly, the task manager 328 receives task requests from the client computer system software 308 and is involved in executing those tasks. The tasks generally relate to actual management or configuration type tasks, such as adding new users to the network, but may also refer to other tasks such as providing task related information to the user. The task manager 328 may perform tasks by interaction with one or more back-end resources 306. To carry out a requested task, the task manager 328 has instructional information stored in task store 340 used to recognize the functions to be performed in carrying out a specific task, such as, for example, which resources must be notified or used in performing the task. For instance, in adding a new user to the network, the task manager 328 may need to send new user information to an employee database managed by a SQL server, to an email server and to a security-clearance application, among others. In such a case, the task manager 328 may store, in task store 340, the set of application or resources that must be notified of the fact that a new user is being added. The task manager 328, the search manager 324 and other managers are discussed in more detail below as they relate to the actual management of the resources 306.

Please replace the paragraph beginning on page 21, line 4 and ending on page 21, line 15 with the following rewritten paragraph:

A15 Initially, flow 400 begins as receive operation 402 receives a request indicating that a resource or "plugin" is requesting to be installed on the system. Resources that hook themselves into the management module 304 are generally referred to as plugins, where the plugin is the portion of the resource that communicates with the management module 304 by sending and receiving messages. The request may be made by the resource itself or the system may recognize that a new resource is being plugged into the system, thus generating the request. In an embodiment the configuration manager 330 (Fig. 3) receives the request. The request, as well as other provided information, is in a predetermined format, such as in XML, so that the management module 304 understands the configuration information. The request typically

A15

includes some predetermined information, such as the name of the resource, the location of various XML files, among others.

Please replace the paragraph beginning on page 22, line 10 and ending on page 22, line 19 with the following rewritten paragraph:

A16

As an example, an Active Directory plugin may exist on the system and it may expose objects relating specific users, known in a preferred embodiment of the invention as user objects. Subsequently, an email application may be installed that recognizes user objects, and specifically provides an email address attribute to existing user objects. Furthermore, the email application may provide available actions to user objects, such as editing email addresses, sending email messages, activating or deactivating an email account, etc. These attributes and actions relating to existing objects are communicated to the configuration manager 330 during receive operation 402. As discussed below, receiving such additional information provides the ability to extend the user object to include the additional information.

Please replace the paragraph beginning on page 22, line 20 and ending on page 23, line 1 with the following rewritten paragraph:

A17

Upon receiving information from the plugin, evaluate operation 404 checks the information in the provided directory to ensure that the plugin is installed correctly. In an embodiment, the configuration manager 330 makes sure that the plugin is installed with the correct XML file formats and validates the web service interfaces that the plugins need to expose to the system ~~304~~ 302.

Please replace the paragraph beginning on page 23, line 16 and ending on page 23, line 22 with the following rewritten paragraph:

A18

Once the information has been supplied to the various managers, store operation 408 stores relevant information in local store(s). That is, each manager that receives information related to a newly installed and registered plugin stores some of the information, if relevant, in its local store. For instance, the task manager 328 may evaluate the new information and determine that some task-related information should be stored in the local store corresponding to the task

A18 manager 328. As stated above, each manager may maintain a dedicated store of information for this purpose.

Please replace the paragraph beginning on page 23, line 23 and ending on page 24, line 2 with the following rewritten paragraph:

A19 Following store operation 408, the plugin is considered to be installed and registered with the management module 304 such that future management can be controlled through the management module 304 and, therefore, the flow 400 ends at end operation 410.

Please replace the paragraph beginning on page 24, line 3 and ending on page 24, line 16 with the following rewritten paragraph:

A20 Although the flow 400 generally relates to adding a new resource to the system, it also relates to extending existing objects within the system. For example, assume the flow 400 is installing an email application to an existing system having an "Active Directory" application that exposes user objects. In this case, since the email application supports user objects, the supply operation 406 essentially supplies new user-object information to managers within the system and the store operation 408 stores the information for later use. The new user object information may include task information, e.g., creating a new email account, editing an email address, activating and deactivating accounts, etc. to the task manager 328. Similarly, if the email address is a searchable attribute, then this information may be communicated to the search manager 324. Essentially, the user object supported by the Active Directory is now also supported by the email application. More importantly, combining information from separate resources into a single, exposed object, provides a relatively comprehensive set of information about each user, such as all attributes and/or actions by simply accessing the object.

Please replace the paragraph beginning on page 25, line 4 and ending on page 25, line 14 with the following rewritten paragraph:

A21 As stated above, in a particular embodiment, the addition of new resources is handled by the configuration manager 330 (shown in Fig. 3), which communicates with resources as they are added to the system. ~~Preferable~~ Preferably the resources are installed on the system and then registered with the management module 304. The configuration manager 330 may further

configure the resources 306 to allow management of those resources, or at least perform a test operation to make sure the resources are installed in compliance with the necessary minimum requirements to allow other managers to access and manage the resource. For instance, as a new resource is added to the environment 300, configuration manager 330 comprises the software element that communicates with that resource in order to install the resource within the system and evaluate whether the resource is properly installed.

Please replace the paragraph beginning on page 28, line 18 and ending on page 29, line 4 with the following rewritten paragraph:

With respect to certain aspects of the present invention, the property sheets that are exposed to the system 304 by one resource are extendible by other resources. Fig. 5 illustrates the concept of having a separate, independent application or resource extend an existing property sheet. In Fig. 5, a property sheet representing a particular user object is illustrated in display 500. Consequently, the display 500 represents the object itself. The object provides a title bar 502, an active region 504, and a scope list 506. The title bar 502 displays the title of an object as defined by the property sheet. The active region 504 displays controls and data fields for one property page, as selected from the list 506. The list 506 lists the various property pages that may be displayed in the active region 504 that relate to the user object 500.

Please replace the paragraph beginning on page 29, line 2 and ending on page 29, line 10 with the following rewritten paragraph:

Initially, the flow 800 begins as a receive operation 802 receives a request to display a task list for a selected object. That is, prior to receive operation 802, a user or the system selects an object and conducts a request indicating that the associated tasks for that object should be collected and displayed. For example, an administrator may highlight a particular user object. Highlighting or otherwise selecting the user object may, in one embodiment, automatically causes a request to be made for the tasks associated with that object. The request to collect and display all the tasks for the object typically includes the object type and the information about the instance or context of the object.

Please replace the paragraph beginning on page 33, line 3 and ending on page 33, line 5 with the following rewritten paragraph:

Using the methods shown in Table 4, the management module 304, and in particular, a property sheet manager 332, may get data from a plugin or resource 306 and modify data that is stored in the resource with respect to property pages.

Please replace the paragraph beginning on page 34, line 12 and ending on page 34, line 16 with the following rewritten paragraph:

The append operation 606 appends the received property page to the property sheet. Essentially, the property sheet definition is modified, such as by the property ~~page~~ sheet manager 332 to include a pointer to the new property page. Thus, the next time the property sheet is called, the new property page information is displayed along with other property pages for the supported object.

Please replace the paragraph beginning on page 37, line 6 and ending on page 37, line 18 with the following rewritten paragraph:

Upon determining which resources must be accessed, the task manager 328 communicates with those resources, such as resources 318, 320 and 322 through a SOAP proxy and a firewall, as shown in Fig. 7. SOAP is an XML-based protocol that is designed to exchange structured and typed information on the Web. The purpose of SOAP is to enable rich and automated Web services based on a shared and open Web infrastructure. SOAP can be used in combination with a variety of existing Internet protocols and formats including HTTP, SMTP, and MIME and can support a wide range of applications from messaging systems to RPC. Alternatively, other proxy communications may be used to provide the communication protocol. The resources are therefore called by the task manager 328 and provided information to carry out specific tasks. Importantly, more than one resource may be called in response to a single task request from the user. The method of combining multiple functions into a single task may be customized by the user as described below with respect to scripting.

Please replace the paragraph beginning on page 43, line 2 and ending on page 43, line 8 with the following rewritten paragraph:

In an embodiment of the invention, the search manager 324 uses a schema driven search method which greatly enhances the flexibility of traditional searching capabilities by allowing the search parameters to be fully configurable through XML schema by the resources themselves. In essence, a resource may choose what object attributes can be searched over and in what domains the search is to be performed while still maintaining the uniformity of the user-interface and the architectural interface across different renderings of the management module 304.

A27

Please replace the paragraph beginning on page 43, line 9 and ending on page 43, line 14 with the following rewritten paragraph:

A28

More particularly, in an embodiment the plugin may provide such information to the configuration manager 330 at process step 402 described above in conjunction with Fig. 4. The plugin provides the configuration manager 330 with a list of all attributes that define their objects and also a list of which of these attributes that may be used in subsequent searches. Additionally, the plugin may indicate the available scopes for such searches. The search manager 324 is then provided this information at supply operation 406.

Please replace the paragraph beginning on page 46, line 10 and ending on page 47, line 5 with the following rewritten paragraph:

A29

In a particular embodiment, the search manager 324 may provide two options to the user, a quick search and an advanced search. With respect to the advanced search function, the user may supply an advance-search query to the search manager 324, which, in turn, parses the query. Based on the results of the parsing action, the search manager 324 may then access the various resources and subsequently search information stored in conjunction with those resources to locate the requested information. Many different advanced search algorithms may be implemented to perform the search function. The search manager 324, however, controls the searches performed, either using plugin search engines, such as engine 1002 or a search engine located on the resource management module 304, not shown. Results from searches may be marshaled back to the user through a search management service 1004 and displayed in the user interface 1006, as described below.

Please replace the paragraph beginning on page 47, line 6 and ending on page 47, line 13 with the following rewritten paragraph:

A30

Fig. 11 illustrates an operational flow 1100 related to the execution of a search function. Initially, a receive operation 1102 receives a search query. The query may be received from the client system ~~308~~ 302. However, in other embodiments, the system 304 may receive the query from other sources, such as one of the managers, e.g., the task manager 328 or one of the resources 306. The query includes information related to the types of objects or tasks that are requested. However, in other embodiments, the query may include any information that can be used to conduct a search either locally on the management module 304, or on one or more resources 306.

Please replace the paragraph beginning on page 49, line 14 and ending on page 49, line 21 with the following rewritten paragraph:

A31

As also described above an application, such as the ~~browser~~ graphical user interface (GUI) 116, may access the web service of the interface manager 326 using web protocols and data formats, such as HTTP, XML, and SOAP. During an initial access of the interface manager 326 by the ~~browser~~ GUI 116, the interface manager 326 downloads a small application, herein referred to as the console applet 313, to the web browser 311, as shown in Fig. 3. The console applet 313 may comprise, for example and without limitation, a Java applet. The console applet 313 operates within the browser to render a management console, within the browser 311, in accordance with predefined console layout specifications and features. As also described below, the console 313 applet also manages the sending and receiving of information to and from the interface manager 326.

Please replace the paragraph beginning on page 49, line 22 and ending on page 50, line 8 with the following rewritten paragraph:

A32

In one embodiment of the present invention, the layout of a format of the particular management console being rendered in the browser 311 on the client computer system ~~116~~ 302 is specified by an XML document sent from the interface manager 326 to the console applet 313. In this embodiment, the console applet 313 then opens the XML file by loading the XML file into an XML Document Object Model (DOM) object. The DOM object is then used to access

the data defining the format of the consol. The console applet 313 then interprets the data and A32 generates appropriate HTML, DHTML, or scripts (such as java script or jscript) for rendering the management consol in the browser 311.

Please replace the paragraph beginning on page 50, line 17 and ending on page 50, line 23 with the following rewritten paragraph:

A33 Additionally, data relating to requests or inquiries from the client environment 302 to the management module 304, or responses from the management module 304 to the client ~~environment~~ computer system 302, may also be formatted and transferred between the client ~~environment~~ computer system 302 and the management module 304 as XML documents. Once the interface manager 326 has received a request or inquiry from the client ~~environment~~ computer system 302, the interface manager 326 functions to distribute the request or inquiry to the appropriate manager or resource.

Please replace the paragraph beginning on page 51, line 1 and ending on page 51, line 7 with the following rewritten paragraph:

A34 The format of the XML documents sent between the console applet 313 and the interface manager ~~316~~ 326 are predetermined. In one embodiment, the format of any XML document sent between the interface manager 326 and the console applet 313 is specific to the particular manager that will ultimately access or handle the data contained in the XML document. For example, as described above, information or requests that are to be handled by the search manager 324 may be transmitted in an XML document that has a format that is uniquely configured for search specific data and commands.

Please replace the paragraph beginning on page 51, line 8 and ending on page 51, line 13 with the following rewritten paragraph:

A35 As shown in Fig. 3, the client ~~environment~~ computer system 302 may include a ~~web~~ browser 311, such as ~~browser~~ GUI 116 shown in Fig. 1, as well as an applet 313 running on the ~~web~~ browser 311. The applet 313 functions to generate a management console, as described below, in the ~~web~~ browser 311. The ~~web~~ browser 311 may be any standard web browser that is compatible with the ~~Microsoft~~ Microsoft® ".NET Framework."

Please replace the paragraph beginning on page 51, line 13 and ending on page 51, line 17 with the following rewritten paragraph:

A36

As is known, a web browser is a client application, software component, or operating system utility that communicates with server computers via standardized protocols such as HTTP, FTP and Gopher. Web browsers receive documents from the computer network and present them to a user. ~~Microsoft~~ Microsoft® Internet Explorer, available from Microsoft Corporation, of Redmond, Wash., is one example of a web browser.

Please replace the paragraph beginning on page 51, line 18 and ending on page 52, line 3 with the following rewritten paragraph:

A37

Fig. 12 illustrates some of the elements of a user interface in accordance with an embodiment of the present invention. As described above, in an embodiment, the user interface comprises what is referred to herein as a management console 1200, which is rendered in a window of a web browser, such as ~~web~~ browser 311 (Fig. 3). In one embodiment, the rendering of the console 1200 may be handled by the web browser ~~1211~~ 311 in accordance with a markup language document, such as, without limitation, an XML document, an HTML document, or a DHTML document. In the case where the consol is rendered in accordance with an XML document, that document may be converted to, and accessed in the form of, a DOM object.

Please replace the paragraph beginning on page 52, line 15 and ending on page 53, line 3 with the following rewritten paragraph:

A38

As shown in Fig. 12, the console 1200 includes a tool bar 1210 and three zones, including a first tool zone ~~412~~ 1212, a work zone 1214, and second tool zone 1216. In one embodiment, the toolbar 1210 is located at an uppermost edge 1202 of the console 1200. It will be appreciated by one skilled in the art that the present invention is not limited to the toolbar described herein, but may encompass any type of toolbar containing control elements or commands for controlling the features of console 1200. The toolbar 1210 may include any number of controls that are associated with the console 1200. In one embodiment, as shown in Fig. 12, the toolbar 1202 includes a console selection element 1222, a show element 1224, a first zone display element

A37 1218, and a second zone display element 1220. These controls perform specific functions in association with the console 1200, as will now be described.

Please replace the paragraph beginning on page 54, line 7 and ending on page 54, line 18 with the following rewritten paragraph:

A38 Additionally, located in the drop-down menu 1332 of the console selection element 1222 are a console "save" element 1334 and console "save as" buttons 1336. The console "save" element 1334 may be used to save the current state of the console layout 1200, under the current console name. In contrast, the console "save as" element 1336 may be used to save the current state of the console 1200 under a different console name. In one embodiment, the console "save" element 1334 and the console "save as" ~~1336~~ element 1336 may be implemented as buttons located in the drop-down menu 1332, as shown in Fig. 13. Once the saved, the layout of a console may be save, for example as an XML file, by the persistence manager 334. The selection of these elements may then be accomplished by "clicking" the buttons using a mouse. However, the console "save" element 1334 and the console "save as" element 1236 may also, or alternatively, be located in other areas of the console 1200.

Please replace the paragraph beginning on page 55, line 3 and ending on page 55, line 12 with the following rewritten paragraph:

A39 In addition to the "hide left zone" element 1342 and the "hide right zone" element 1344, in one embodiment, the show drop-down menu 1340 may also include other elements for selecting or deselecting the display of the various tools located in the first zone 1212 and/or the second zone 1216 of the console 1200. For example, as shown in Fig. 13, the show drop-down menu 1340 includes a "hide quick search tool" element 1346 and a "hide monitors tool" element 1348. The selecting or deselecting of the display of various the various tools located in the first zone 1212 and/or the second zone 1216 of the console 1200 may be accomplished in a similar manner as that described above with respect to the "hide right zone" element 1344 and the "hide left zone" element 1342.

Please replace the paragraph beginning on page 56, line 7 and ending on page 56, line 13 with the following rewritten paragraph:

In an embodiment of the present invention, the first zone 1212 and the second zone 1216, are operable to display one or more tools, where a tool is a graphical user interface element that provides a user quick access to features or functions of the console 1200. For example, as shown in Fig. 12, the first zone includes a quick search tool 1240, which provides a hierarchical selection structure to enable the user to quickly search for different objects and to populate the work zone 1214. As also shown in Fig. 12, the second zone includes a monitor tool that displays the status of CPU usage.

A40

Please replace the paragraph beginning on page 56, line 22 and ending on page 57, line 10 with the following rewritten paragraph:

A41

Located within the workspace is workspace window 1252, which is operable to display one or more modules 1254. Also included in the workspace 1250 is a scroll bar 1255. The scroll bar 1255 is a graphical control element that allows a user to view information outside of the viewing area of the workspace window 1252. For example, the scroll bar 1255 may be used to scroll the workspace window 1252 in a manner that brings one or more ~~another module~~ of the modules present in the workspace 1250 into view in the workspace window 1252. As is typical, the scroll bar 1255 includes a scroll box 1256 that may moved up and down inside the scroll bar 1255 using a mouse. Scroll arrows 1258 at each end of the scroll bar 1255 can also be clicked to move the viewing area of the workspace window 1252 in a specified direction. Additionally, the scroll box 1256 may be moved up and down inside the scroll bar 1255 by clicking inside of the scroll bar 1255 in an area not occupied by the scroll box 1256, as is conventional.

Please replace the paragraph beginning on page 57, line 11 and ending on page 57, line 16 with the following rewritten paragraph:

A42

Located within the workspace window 1252 are one or more modules 1254. Where a workspace is mapped to job functions, modules 1254 typically map to a specific object(s) upon which work is done. As discussed above, there are two types of distinct modules—those which reside in a workspace and pertain to a specific object type and those which reside in the first and second zones, herein referred to as tools. Table 8, shown below, illustrates a list of several modules that may be displayed within the various zones.

Please replace the paragraph beginning on page 59, line 1 and ending on page 59, line 8 with the following rewritten paragraph:

A43

In general, the object pane 1264 is operable to display information about one or more objects that are applicable to the module 1260 in which the object pane 1264 resides. For example, the object pane 1264 may include a list of objects associated with a given module that may be selected for access by a user. The objects in the object pane 1264 may be presentation in a number of ways, depending on the number of objects to be displayed and the preferences of the author of the console 1200. For example, and without limitation, the objects may be displayed in a simple object list, as shown in Fig. 14. A user may then select one or more of the objects in the list for accessing.

Please replace the paragraph beginning on page 59, line 9 and ending on page 59, line 15 with the following rewritten paragraph:

A44

The task pane 1266 is operable to display various tasks that are or applicable to, and available for, an object that has been selected in the object pane 1264. As with the object pane 1264, tasks in the task pane 1266 may be presentation in a number of ways, depending on the number of tasks to be displayed and the preferences of the author of the console 1200. For example, and without limitation, the tasks may be displayed in the task pane 1266 as a simple task list, as shown in Fig. 14. A user may then select one or more of the tasks in the list for access.

Please replace the paragraph beginning on page 59, line 16 and ending on page 59, line 22 with the following rewritten paragraph:

A45

Once a task has been selected from the task list ~~1266~~ 1014, the function of the selected task may be immediately carried out or, alternatively, a work pane 1268 may be displayed showing additional information and/or presenting additional functionality, or sub-tasks, related to the selected task. The additional information, the presented additional functionality, and/or the related sub-tasks, may be displayed in the work pane 1268 in a number of ways, depending on the type of information or functionality that is to be displayed and the preferences of the author of the console 1200.

Please replace the paragraph beginning on page 60, line 17 and ending on page 60, line 21 with the following rewritten paragraph:

A46

In one embodiment, the result of executing a query produces one of three states: 1) no objects found; 2) one object found; or 3) multiple objects found. If an object or multiple objects are found, they are displayed in the objects pane 1264. As such, refreshing or modifying the query may result in different objects being displayed in the object pane 1264.

Please replace the paragraph beginning on page 60, line 22 and ending on page 61, line 5 with the following rewritten paragraph:

A47

The query pane 1262 provides a unique and cohesive approach to object selection and management. Instead of requiring the network administrators to navigate to an application and then navigate to an object or group of objects, as was common in prior network administration tools. The query pane 1262 allows a network administrator to navigate directly to an object or group of objects. Once an object(s) is located the network administrators is then able to perform all task that are associated with that object(s).

Please replace the paragraph beginning on page 62, line 2 and ending on page 62, line 11 with the following rewritten paragraph:

A48

As shown in Fig. 15, the quick search tool ~~618~~ 1418 employs common GUI controls such as the drop-down menus, text boxes, and buttons. The controls are arranged logically to support a simple work flow for performing the action of specifying an object instance-action pair to be found. In an embodiment, the quick search tool ~~618~~ 1418 includes a tool bar 1510, including a quick search drop-down menu selector 1512, an edit element 1514, and a quick search close element 1516. The quick search drop-down menu selector 1512 includes a triangular visual element that may be "clicked" on by a mouse to open or close a quick search drop-down menu 1518, in a conventional manner. The quick search close element 1516 includes an x-shaped visual element that may be "clicked" on by a mouse to close the quick search tool ~~1518~~ 1418, in a conventional manner.

Please replace the paragraph beginning on page 66, line 10 and ending on page 66, line 17 with the following rewritten paragraph:

A49

Once a user has selected an object type, scope, instance(s), and action, the search is initiated using quick search initiation element 1538. The quick search initiation element 1538 may be displayed in a number of ways. For example, and without limitation, the quick search initiation element 1538 may display as a search button, as shown in Fig. 19. To initiate the search, the user may then click on the search initiation element 1538 or "GO!" button ~~1538~~. In the example shown in Fig. 19, the search button is labeled "Go!" However, it will be understood that other labels are possible to indicate the function of the search ~~button~~ initiation element 1538 (e.g., Search, Begin, Start, OK, etc.).

Please replace the paragraph beginning on page 66, line 22 and ending on page 67, line 7 with the following rewritten paragraph:

A50

When the user selects the GO button 1538, a search string is generated and embedded in an XML document described above with respect to Table 7. The XML document is passed to the search management service (an asmx file that exposed the search manager object) and then passed directly to the search manager 324. The search manager 324 parses the query string in the XML document to obtain the object type identification. The search manager 324 then searches an object type database and obtains the XML schema for the provided object type, such as during the determine operation 1106 described above. The search manager 324 then inserts the query string from the query string XML document into an attribute designated as quick searchable.

Please replace the paragraph beginning on page 69, line 4 ending on page 69, line 13 with the following rewritten paragraph:

A51

Fig. 21 illustrates an enlarged view of the explorer tool 1420 shown in Fig. 14. As shown in Fig. 21, in one embodiment, the explorer tool 1420 includes a tool bar 2108, having an explorer drop-down menu selector 2110, an edit element 2112, and a explorer close element 2114. The explorer drop-down menu selector 2110 includes a triangular visual element 2116 that may be "clicked" on by a mouse to open or close an explorer drop-down menu ~~1318~~ 2110, in a conventional manner. The explorer close element 2114 comprises an x-shaped visual element

that may be "clicked" on by a mouse to open or close the explorer tool ~~1412~~ 1420, in a conventional manner. In one embodiment, the edit element 2112 provides functionality for a user to add, delete, and/or move workspaces within the explorer.

Please replace the paragraph beginning on page 73, line 19 and ending page 74, line 9 with the following rewritten paragraph:

In response to the selection of the particular objects 2320 and 2334, a list of applicable tasks 2322 has been displayed in the task pane 2314. Included in the list of applicable tasks 2322 is a compare properties task 2332 that has been selected in the task pane, as shown by highlighting. In response to the selection of compare properties task 2332, a section of the work pane 2316 has been populate with a property sheet 2326 associated with the object 2320 titled "Kristy Wallace." Included in the property sheets 2326 is a list of property pages 2328 that are associated with the property sheet ~~3326~~ 2326. Included in the list of property sheets 2326 is a general property page 2324 that has been selected in the list of property pages 2328, as shown by highlighting. In response to the selection of general property page 2324, the work pane 2316 has been populated with a general property page 2330 associated with the "Kristy Wallace" object 2320, as well as a general property page 2340 associated with the "Tim Jones" object ~~2320~~ 2334. Each of the property pages includes a number of controls for editing the general property pages 2330 and 2340.